Automatic question answering

Ottosen, Emil Solsbæk

Publication date:
2019

Automatic question answering

Emil Solsbæk Ottosen, emilo@math.aau.dk

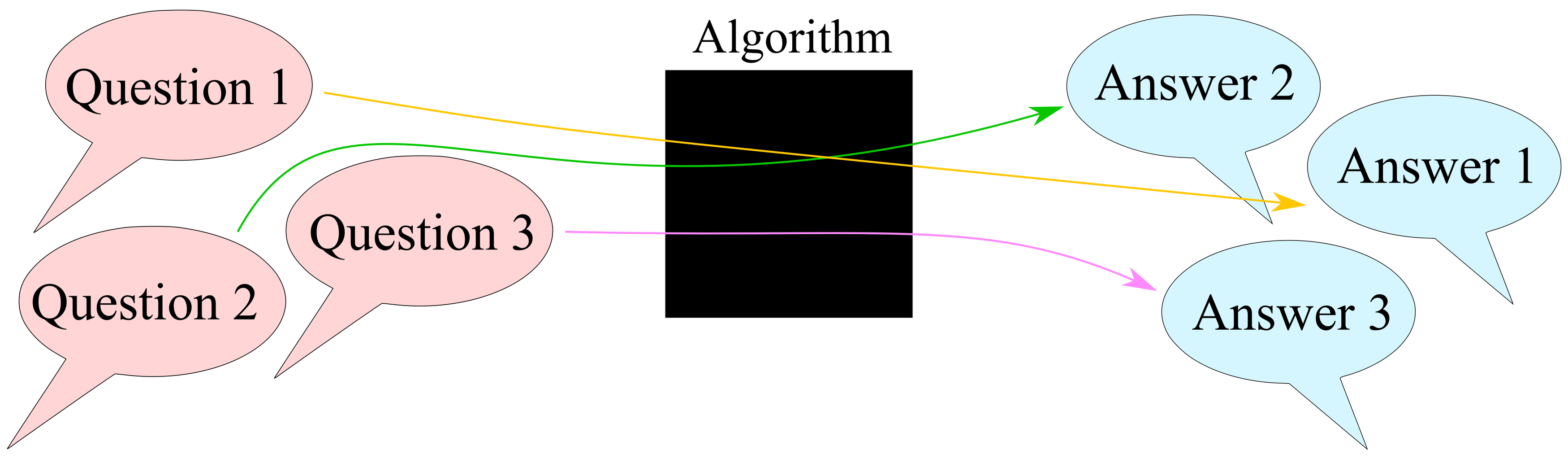
Department of Mathematical Sciences, Aalborg University, Denmark
— in collaboration with Grundfos

Introduction

We investigate the problem of designing and understanding algorithms for automatic question answering. The project is in collaboration with the Danish pump manufacturer Grundfos, who are interested in implementing automated chat-bots as part of their on-line services. A standard approach for designing such algorithms is to use *machine learning*.

Machine learning

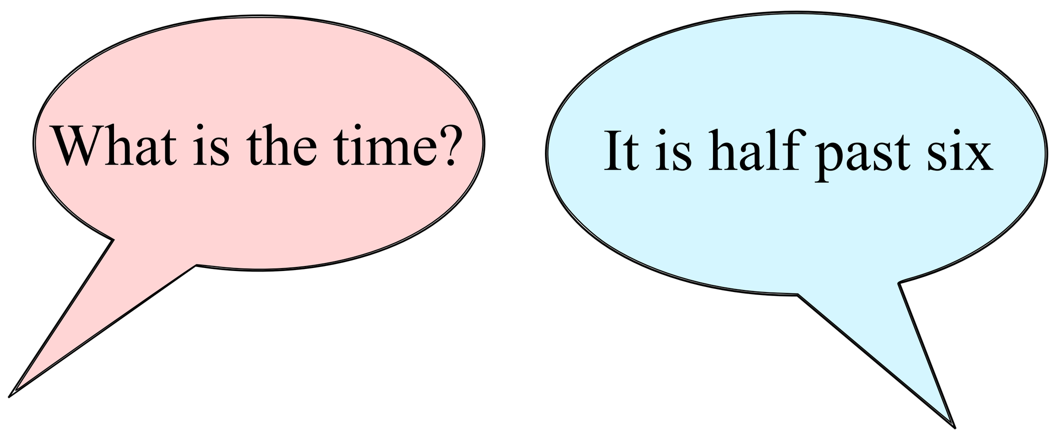
Machine learning is the study of algorithms used to perform a specific task without any explicit instructions. For the particular task of automatic question answering, the computer is given a *labelled training set* consisting of a set of questions and a corresponding set of answers. An algorithm is then *trained* to map each question to the corresponding answer:



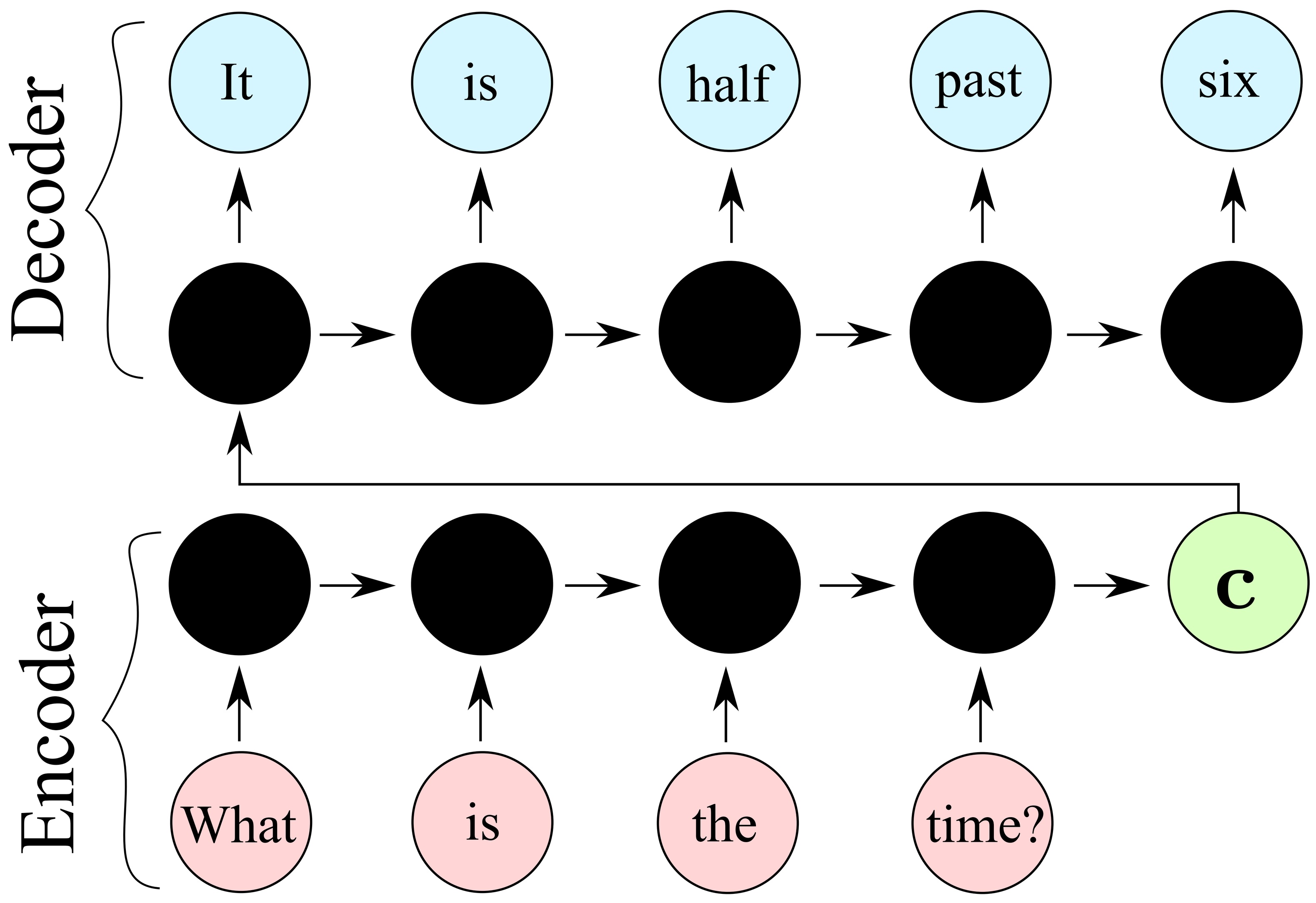
More precisely, the algorithm is trained to produce a plausible answer for each question, which is then compared to the "true" answer using a suitable metric (see [5] for details). We are interested in designing algorithms, which after training can *generalize* beyond the training set, i.e., produce reasonable answers for new questions not contained in the training set. A particular successful approach has been to apply a *neural network* architecture for the algorithms called *encoder-decoder networks* [1].

Encoder-decoder networks

Let us now explain the basic idea behind an encoder-decoder network. Consider the following dialogue:



An encoder-decoder network uses an *encoder network*, which inputs the question and outputs its *context* **c**, which is a (mathematical) representation that summarizes the question. The context is then given as input to a *decoder network*, which produces an answer to the question:



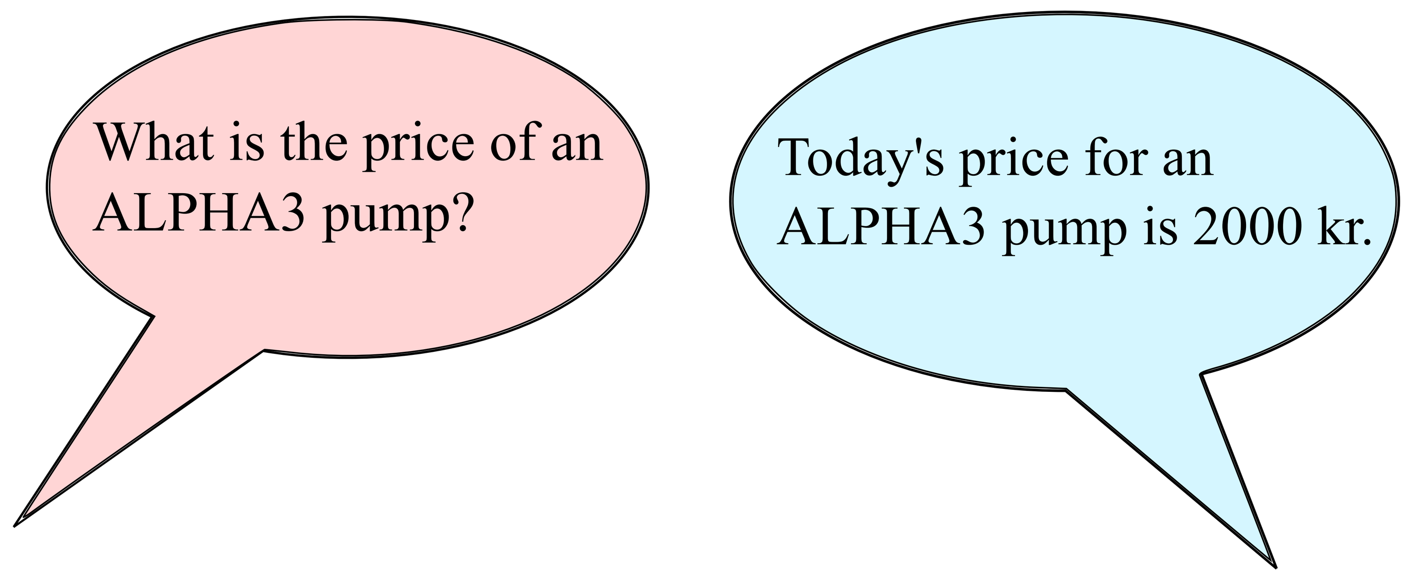
The black circles inside the figure above represent the *hidden cells* inside the encoder network and the decoder network, which are both designed as *recurrent neural networks* (see [2] for details). The choice of color (black) represents the fact the computations inside each cell are very difficult to understand for a human after the algorithm has finished training.

Dataset

In practice, several thousand training examples must be provided for an encoder-decoder network to obtain a performance close to state-of-the-art (which is by no means a perfect performance). Additionally, all answers must of course be provided by humans in order to establish a "ground truth" that the network can learn from. Fortunately, a large company such as Grundfos has many conversations each day with their customers, and these conversations can serve as our training set (after a significant amount of preprocessing).

Problem

We are interested in understanding what is going on inside an encoder-decoder network during and after training. In other words, how can we understand the nature of a trained network, when the computer has never been given any explicit instructions? Consider the following dialogue between a customer and an employee at Grundfos:



If the price of an ALPHA3 pump would be fixed for all eternity, then we could simply train the network to learn this fixed price. Unfortunately, the price of an ALPHA3 pump varies on a daily basis and therefore cannot be learned by the network. This constitutes a problem, since we do not know how the network "understand" the concept of a price. Where does the network store the corresponding information and how does it retrieve it? Is the information contained in a single hidden cell or spread across out the entire network?

Related work

Recent research has shown that trained recurrent neural networks do indeed produce interpretable cells for certain tasks — it is just not clear which cells and for what tasks [3]. Furthermore, it has been proposed in [4] that encoder-decoder networks implicitly decompose sentences into so-called *filler-role bindings*, where the *filler* is a word and the *role* is the position of the word in a sentence (position can then be defined in several ways). To support their hypothesis, the authors approximate the context **c** of an encoder-decoder network using a *tensor product decomposition network*. Then, they construct a hybrid network consisting of the tensor network and the original decoder and compare the performance of this hybrid network against the performance of the original network.

References

[1] K. Cho, B. van Merrienboer, a. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*, 2014.

[2] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[3] A. Karpathy, J. Johnson, and F.-F. Li. Visualizing and Understanding Recurrent Networks. *CoRR*, abs/1506.02078, 2015.

[4] R. T. McCoy, T. Linzen, E. Dunbar, and P. Smolensky. RNNs implicitly implement tensor-product representations. In *International Conference on Learning Representations*, 2019.

[5] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics, 2002.